Ordinateurs, Structure et Applications

Démonstration : Décomposition des instructions en micro-instructions

---

Guide de l'utilisateur

---

7 décembre 2014

Table des matières

1 Description du simulateur

Cette démonstration illustre la décomposition d'instruction Assembleur en micro-instructions simple. Lors de l'exécution de chaque micro-instruction, les lignes d'activation et les liens de communication utilisés par celle-ci sont mis en évidence. Les valeurs des registres du microprocesseur sont mise à jour à chaque pas de simulation.

2 Plateformes supportées

Le simulateur a été testé sur les plateformes suivantes :
Linux : ArchLinux, Fedora 20, Ubuntu 14.10
Mac : Mac OSX 10.8, 10.9 et 10.10
Windows : Windows 7 et 8.1
Le fonctionnement du simulateur sur les autres plateformes n'est pas garanti.

3 Procédure de lancement du simulateur

Le simulateur est fourni avec toutes les librairies nécessaires pour son fonctionnement. Voici la procédure pour rouler le simulateur sur les différentes plateformes.

- Sur **Linux**, il suffit d'extraire l'archive tar.gz et d'exécuter le fichier exécutable s'y trouvant.
- Sur **Mac**, le simulateur est fourni dans le format application compressée (.dmg). Il est possible qu'il soit nécessaire de confirmer l'exécution en l'ouvrant avec CTRL + Click.
- Sur **Windows**,vous devez installer le *Microsoft Visual C++ Redistributable Package*. L'installateur est disponible sur le site suivant : http://www.microsoft.com/en-us/download/details.aspx?id=14632. Ensuite, il suffit d'extraire l'archive zip et d'exécuter le fichier exécutable s'y trouvant.

4 Composantes du microprocesseur

4.1 ALU

Le microprocesseur contient un ALU permettant d'exécuter des additions et des soustractions. L'ALU reçoit deux opérandes sur 16 bits et retourne un résultat sur 16 bits. Il ne possède pas de drapeau.

Lors de l'exécution de l'ALU, si une addition produit une valeur de plus de 16 bits ou si le résultat d'une soustraction est négatif, la valeur retournée par l'ALU se composera des 16 bits les moins significatifs du résultat (overlap). Le bit de signe est ignoré.

Deux multiplexeurs (MUX_A et MUX_B) permettent de sélectionner les valeurs des opérandes. Les bits de sélections des multiplexeurs sont fournis par les lignes d'activations.

Le résultat de l'ALU est inscrit dans le registre accumulateur (ACC).

4.2 Bus

Le microprocesseur contient un bus interne permettant de transférer des valeurs d'un registre à l'autre. De plus, il peut accéder au bus d'adresse en utilisant le registre MAR et au bus de données en utilisant le registre MDR. Le bus de contrôle est directement accessible à l'aide des lignes d'activation de la mémoire de micro-instruction.

Le bus d'adresse contient 7 bits et le bus de données 16 bits.

4.3 Mémoire de micro-instruction

Le microprocesseur possède une mémoire de micro-instruction à l'intérieur de son CCU (Central Control Unit). Cette mémoire contient les décompositions en micro-instruction de chaque instruction une à la suite de l'autre. L'utilisateur ne peut pas modifier son contenu.

4.4 Mémoires externes

Le microprocesseur a accès à une mémoire d'instructions en lecture seule ainsi qu'une mémoire de données disponible en écriture et en lecture.

Les mémoires d'instruction et de données contiennent 64 adresses chacune et contiennent des valeurs sur 16 bits.

Il est possible de charger le contenu initial des mémoires au début de la simulation en fournissant des fichiers de configuration. Si aucun fichier de configuration est fourni, la mémoire d'instruction contiendra des instructions vides à chaque adresse et celle de données la valeur 0x0.

4.5 Registres

Le microprocesseur contient les registres suivants :

— Registres généraux (R0,R1,R2,R3)
— Registre de programme (Program Counter) contenant l'adresse de l'instruction à exécuter
— Registre d'instruction (IR) contenant le code binaire de l'instruction présentement exécutée
— Registre d'adresse mémoire (MAR) contenant la valeur du bus d'adresse
— Registre de donnée mémoire (MDR) contenant la valeur du bus de données
— Registre accumulateur (ACC) contenant le résultat de l'ALU

Tous les registres possèdent des valeurs sur 16 bits, sauf pour le registre MAR qui une valeur sur 7 bits. Si on assigne une valeur de plus de 7 bits au registre MAR, on conserve les 7 bits les moins significatifs de cette valeur.

5 Interface du simulateur

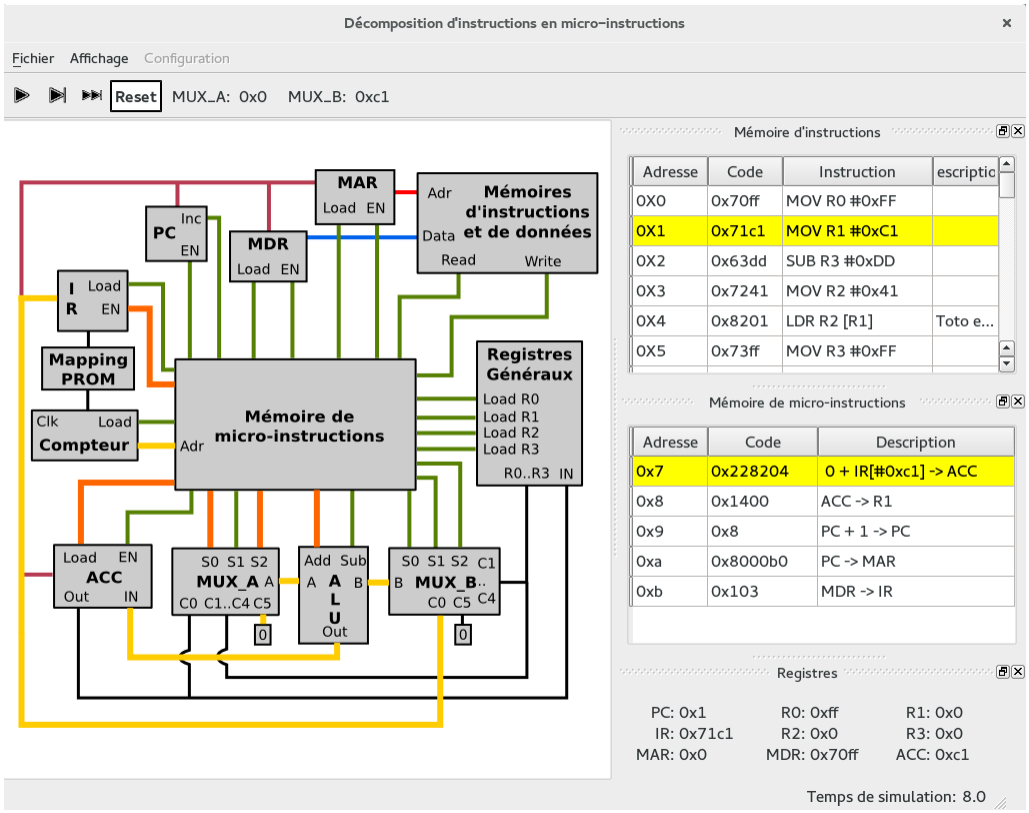

FIGURE 1 – Interface du simulateur

5.1 Mise en évidence des lignes actives

Lors de l'exécution d'une micro-instruction, les lignes d'activation et de communication utilisées sont mises en évidence sur le circuit du microprocesseur. Les lignes d'activation mise en évidence seront de couleur orange et celles de communications en jaune.

5.2 Affichage des valeurs des mémoires et des registres

Une barre d'ancrage à la droite contient les fenêtres affichant les valeurs des mémoires et des registres. Il est possible de détacher chaque fenêtre en cliquant sur le bouton dans le coin supérieur droit de la fenêtre. Lorsque détachée, une fenêtre peut être rattachée dans la barre d'ancrage en double-cliquant dans sa barre de titre.

De plus, il est possible de fermer les fenêtres en cliquant sur le "X" dans le coin supérieur droit de celles-ci. Les fenêtres peuvent être rouvertes à partir du menu "Affichage".

6 Contrôle de la simulation

Le simulateur permet d'exécuter une simulation en continu, pas à pas et jusqu'à la fin. La vitesse d'exécution en seconde entre les rafraîchissements de l'interface et le nombre de pas de simulation effectué par rafraîchissement peuvent être réglés dans le menu principal à partir de l'option "Préférences" sur Mac et "Options" sous Linux et Windows.

Les quatre boutons en haut à gauche de la fenêtre permettent de contrôler le déroulement de la simulation. Voici les fonctionnalités offertes par chaque bouton.

: Permet de réinitialiser la simulation.


: Permet d'exécuter des pas de simulation en continu.


: Permet de suspendre l'exécution de la simulation .


: Permet d'exécuter un pas de simulation.


: Permet d'exécuter tous les pas de simulation successivement.

7 Contenu des mémoires

7.1 Chargement

Le contenu de la mémoire de données et d'instructions peut être chargé à partir du menu "Configuration". Un message d'erreur apparaîtra si le fichier sélectionné n'est pas valide. Après le chargement des fichiers de configuration, le contenu des fichiers est affiché dans les fenêtres détachables correspondantes.

7.2 Contenu initial de la mémoire de données

Le contenu initial de la mémoire de données peut être fourni à l'aide d'un fichier XML. Ce fichier doit utiliser cette structure :

```
<DataMemory>
    <Data Address="0x2f" Value="0xd948" />
    <Data Address="0x27" Value="0x5dc7" />
</DataMemory>
```

À l'intérieur d'une balise nommée *DataMemory*, un élément de type *Data* est ajouté pour chacune des adresses à configurer. Toutes les adresses qui ne possèdent pas d'élément dans le fichier se verront attribuer 0 comme valeur.

7.3 Contenu initial de la mémoire d'instruction

Le contenu initial de la mémoire d'instruction peut être fourni à l'aide d'un fichier Assembleur ayant l'extension .asm . Dans ce fichier, chaque ligne contient une seule instruction Assembleur respectant les

contraintes du jeu d'instructions. Un commentaire peut être ajouté à la fin de chaque ligne en utilisant une apostrophe ('). Un dièse (#) permet d'insérer une constante comme opérande d'une instruction.

Voici un exemple de fichier Assembleur :

```
MOV R0 #0xC7
MOV R1 #0x59
MOV R2 #0xB0
LDR R3 [R0] 'Chargement de la valeur du clavier dans R3
STR R3 [R1] 'Ecriture dans la memoire de donnees
STR R3 [R2] 'Ecriture dans l'ecran
ADD R2 R1
SUB R0 R1
```

8 Jeu d'instruction du micro-processeur

Toutes les instructions du microprocesseur sont sur 16 bits et se décomposent comme suit :

Bits 15 à 12 : Opcode de l'instruction

Bits 11 à 8 : Registre utilisé comme premier paramètre.

Bits 7 à 0 : Registre ou constante utilisés comme deuxième paramètre

Le microprocesseur possède quatre registres généraux nommés R0,R1,R2 et R3.

Le jeu d'instruction supporte les quatre instructions suivantes où Rd est le registre destination et Rs le registre source :

Mnémonique	**Opcode**	**Description**
MOV Rd Rs	0000	Écriture de la valeur du registre Rs dans le registre Rd
MOV Rd Const	0100	Écriture d'une constante dans le registre Rd
ADD Rd Rs	0001	Addition des valeurs des registres Rd et Rs et insertion du résultat dans le registre Rd
ADD Rd Const	0101	Addition de la valeur du registre Rd avec une constante et insertion du résultat dans Rd
SUB Rd Rs	0010	Soustraction de la valeur Rs à l'intérieur de registre Rd.
SUB Rd Const	0110	Soustraction d'une constante à l'intérieur du registre Rd
LDR Rd [Rs]	1000	Chargement d'une valeur se trouvant à l'adresse Rs de l'ordinateur dans un registre.
STR Rd [Rs]	1001	Écriture de la valeur d'un registre à l'adresse Rs de l'ordinateur.

Table 1 – Jeu d'instructions du microprocesseur

9 Composition d'une micro-instruction

Dans ce simulateur, le code d'une micro-instruction contient 25 bits. Chaque bit correspond à une ligne d'activation. Quand une ligne d'activation est utilisée, le bit correspondant est égal à 1. Voici le bit correspondant à chaque ligne d'activation.

Bit	Route
0	Load Compteur
1	Load IR
2	Enable IR
3	Incrémentation PC
4	Enable PC
5	Load MAR
6	Enable MAR
7	Load MDR
8	Enable MDR
9	Load ACC
10	Enable ACC
11	Load R0
12	Load R1
13	Load R2
14	Load R3
15	MUX_A : Bit de sélection 0
16	MUX_A : Bit de sélection 1
17	MUX_A : Bit de sélection 2
18	MUX_B : Bit de sélection 0
19	MUX_B : Bit de sélection 1
20	MUX_B : Bit de sélection 2
21	Addition ALU
22	Soustraction ALU
23	Bus de contrôle : Lecture
24	Bus de contrôle : Écriture

TABLE 2 – Bit appartenant à chaque ligne d'activation